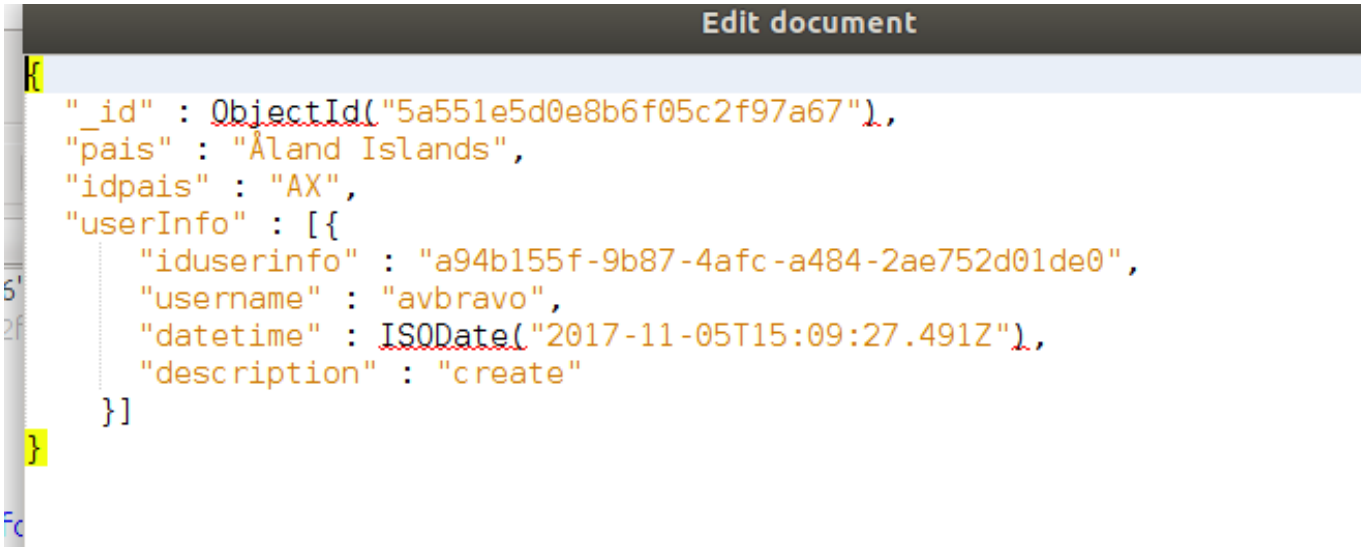
MongoDB

Bases de Datos Avanzadas

Universidad del Rosario

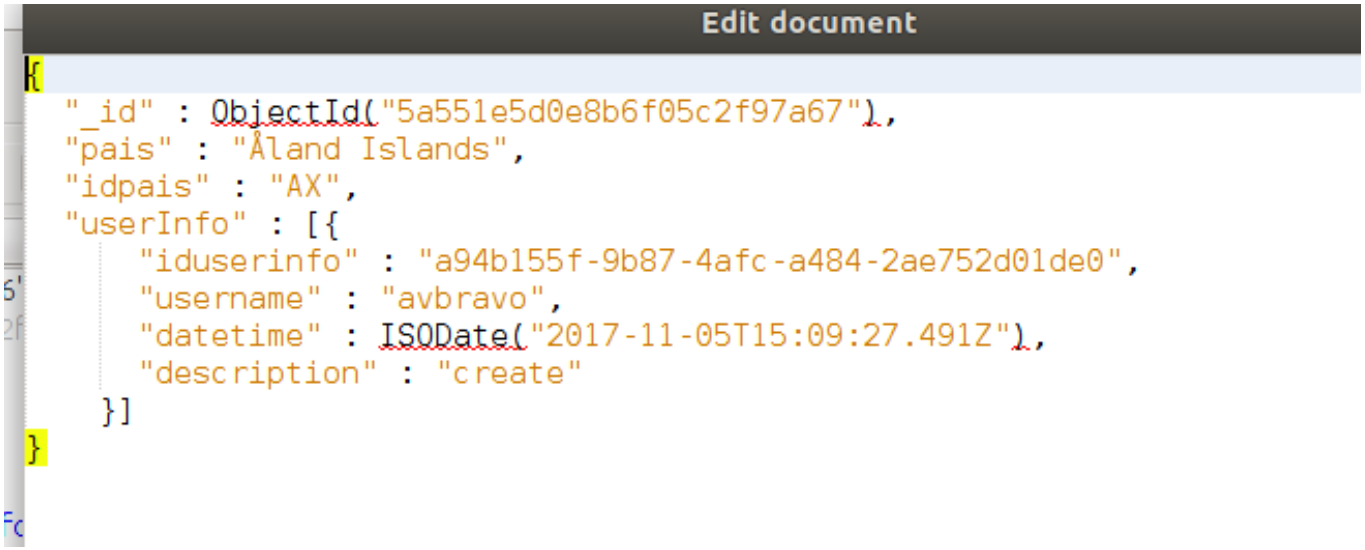
Maestría MACC

Estructura General



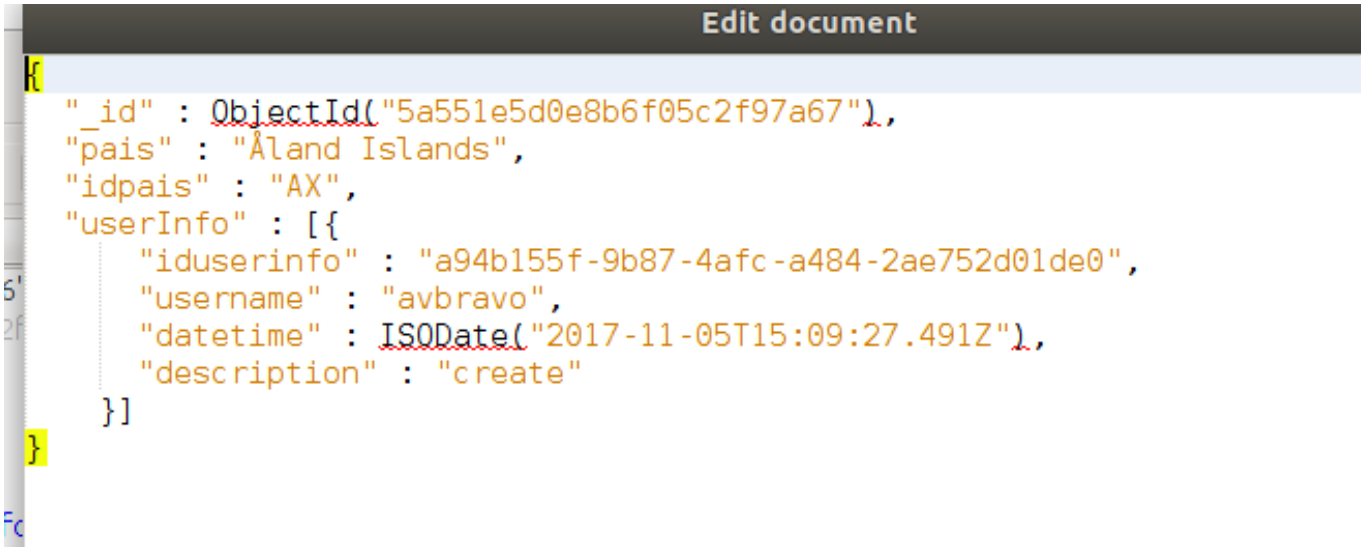
- JSon: *JavaScript Object Notation*.
- Diccionario de datos. Almacenamiento en BSON (*Binary JSon*).
- Sin esquema/*schemaless*.

Estructura General



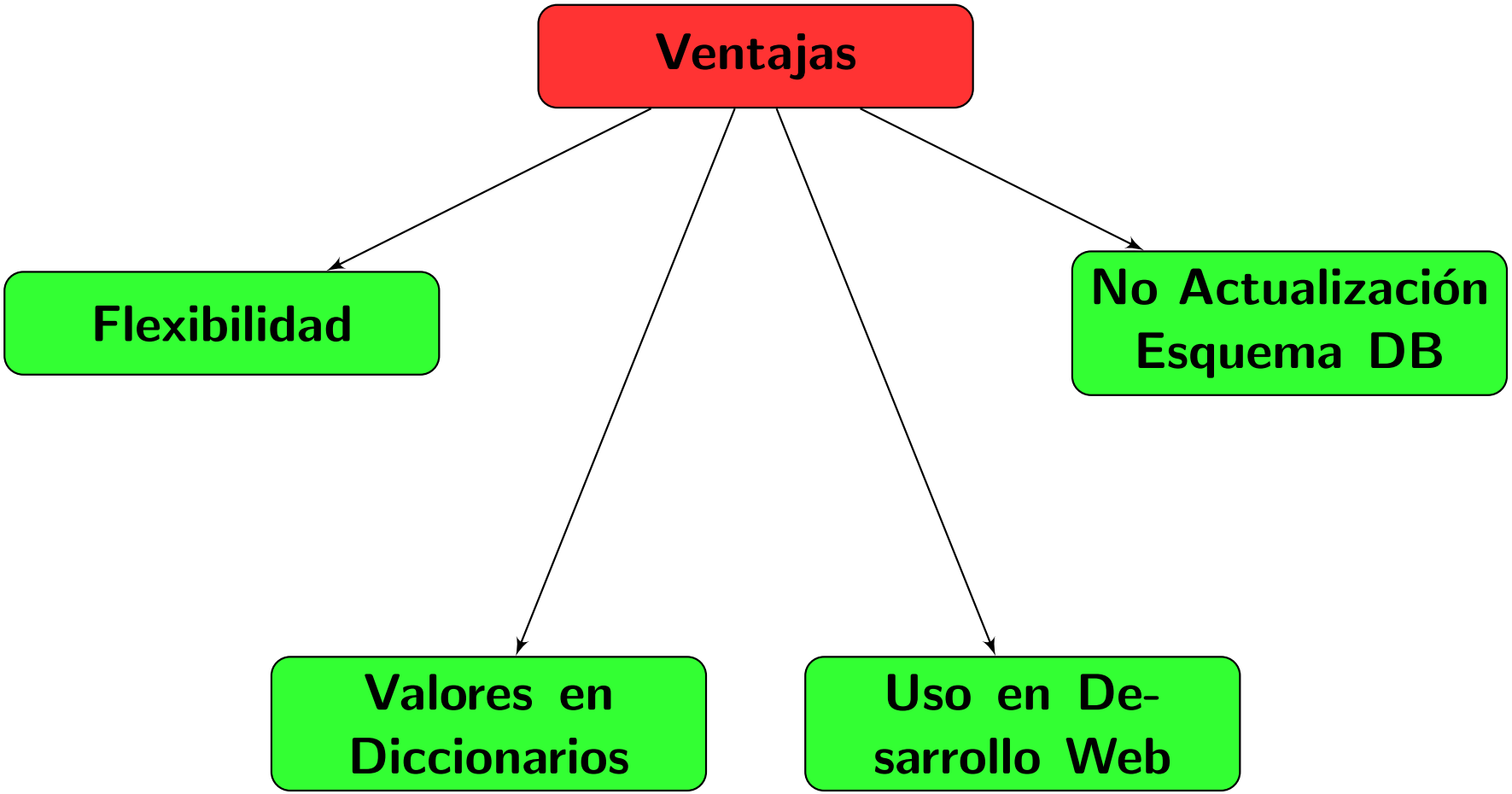
- JSon: *JavaScript Object Notation*.
- Diccionario de datos. Almacenamiento en BSON (*Binary JSon*).
- Sin esquema/*schemaless*.

Estructura General



- JSon: *JavaScript Object Notation*.
- Diccionario de datos. Almacenamiento en BSON (*Binary JSon*).
- Sin esquema/*schemaless*.

Ventajas de JSon



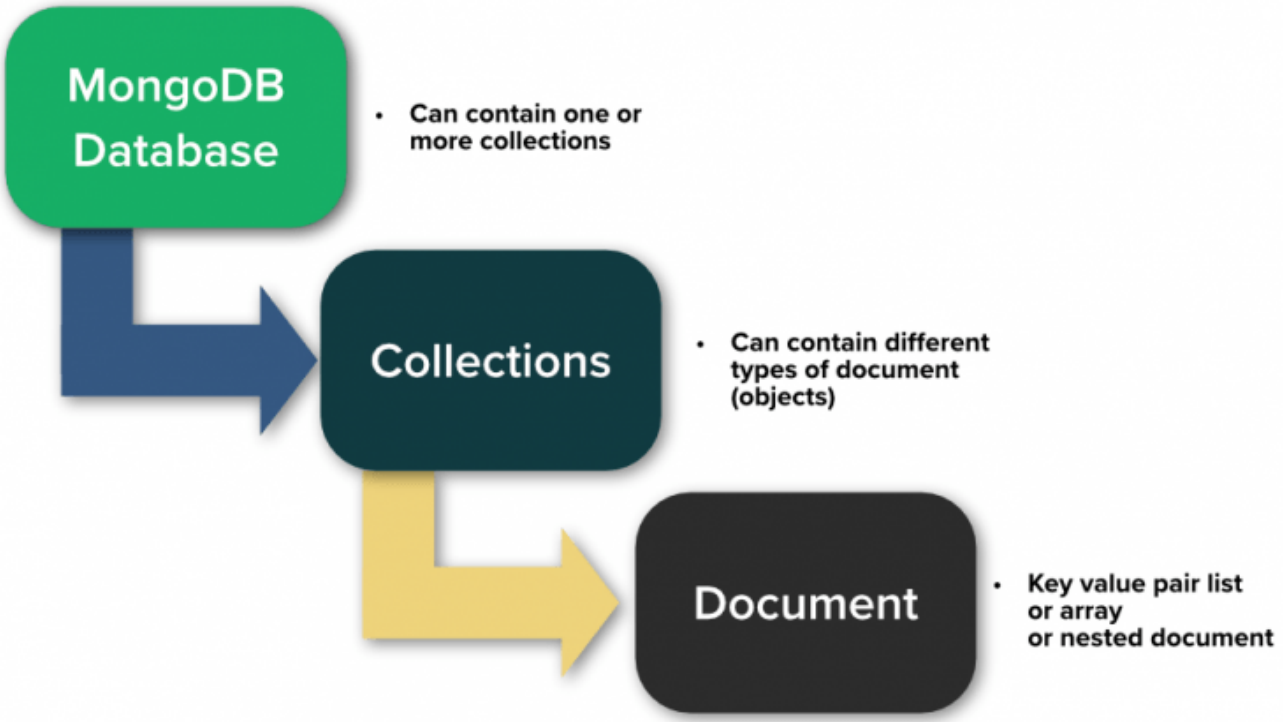
Tipos de Datos en JSon

Tipo de dato	Descripción	Ejemplo
String	Cadena de texto	“Hola”, ’Mundo’
Number	Entero o de punto flotante	42, 3.14
Boolean	Valor booleano	true, false
Date	Fecha y hora	ISODate(”2022-02-27T01:23:45.678Z”)
Object ID	Identificador único	ObjectId(”61ab95d4368a2a523ca011e7”)
Array	Lista de valores	[1, 2, 3], [“A”, “B”, “C”]
Null	Valor nulo	null
Binary Data	Datos binarios	BinData(0, “AQID”)

Diferencias entre JSon y BSon

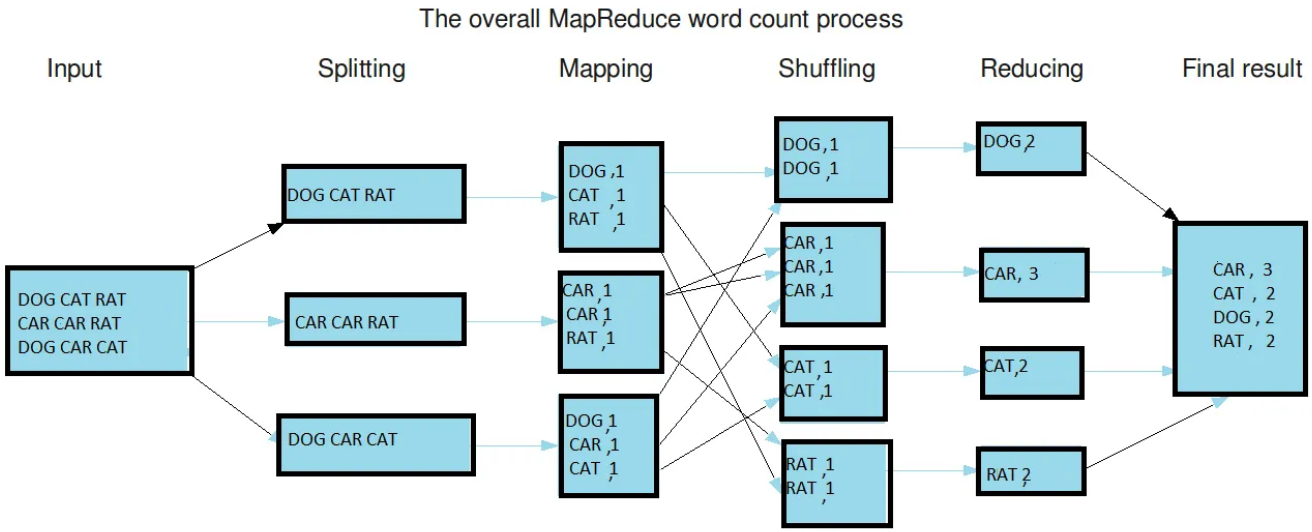
Característica	JSON	BSON
Tamaño	Más grande	Más pequeños
Tipos de datos	Limitados	Amplio
Compresión	No	Sí
Orden de campos	Importa	No importa

Jerarquía de Datos



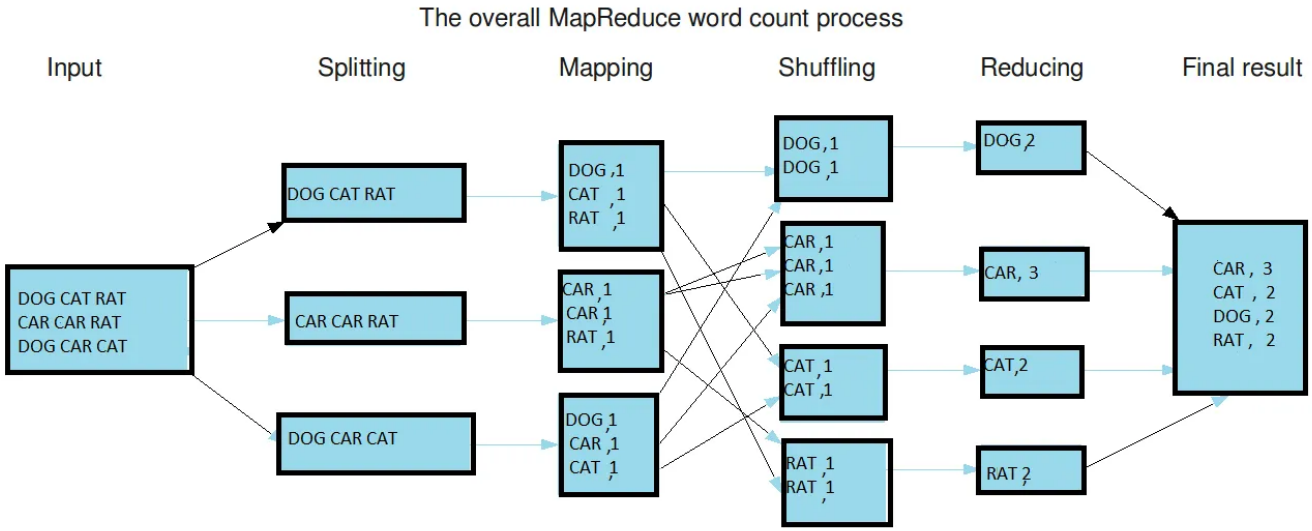
I. Base: **documento**.
II. Colección.
III. Base de Datos Mongo.

Procesamiento *Map-Reduce*

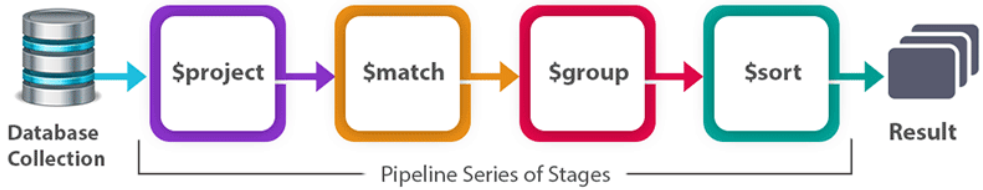


- Soporte para computación en paralelo.
- Actualmente: **pipelines** de MongoDB

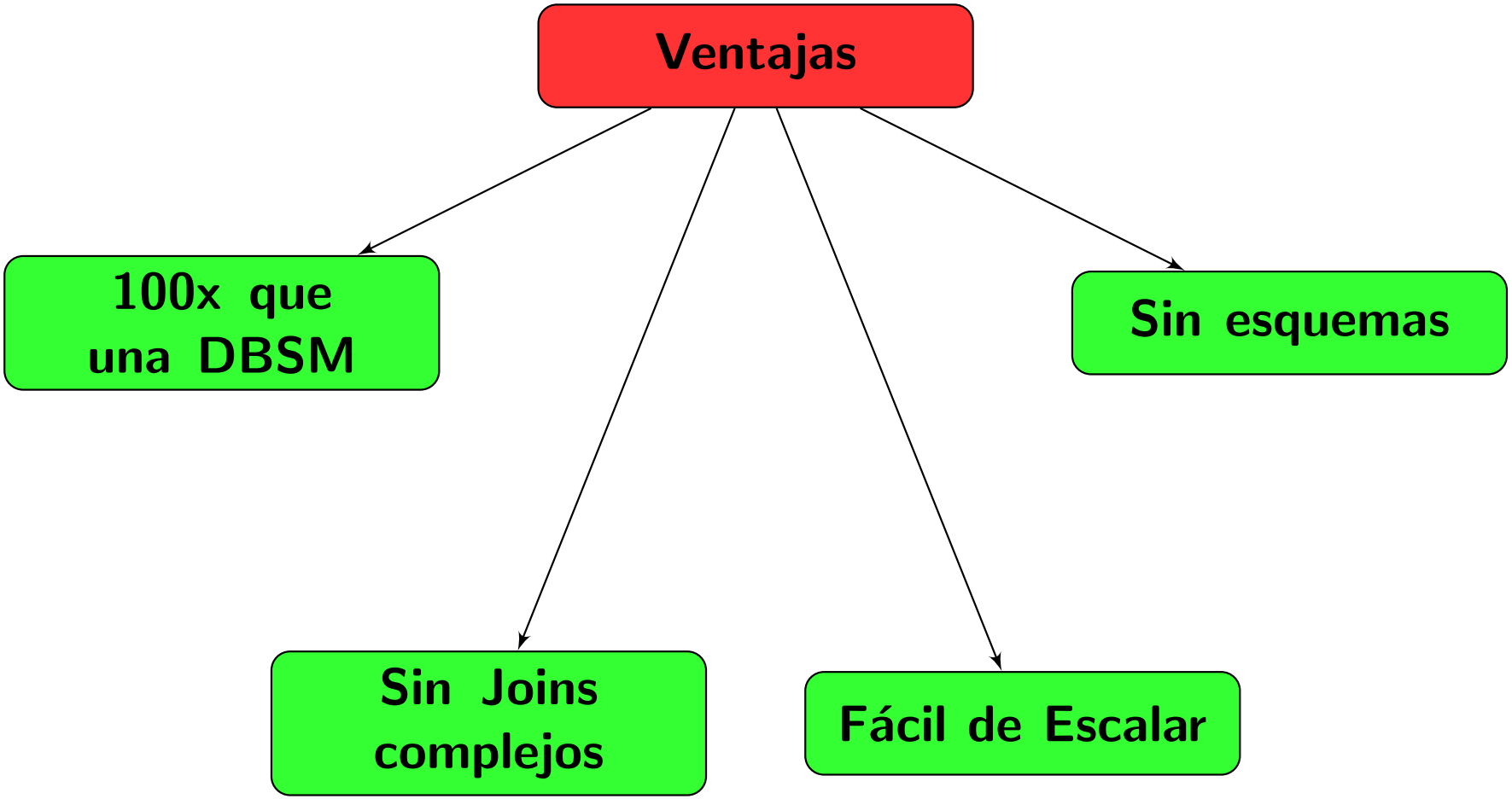
Procesamiento *Map-Reduce*



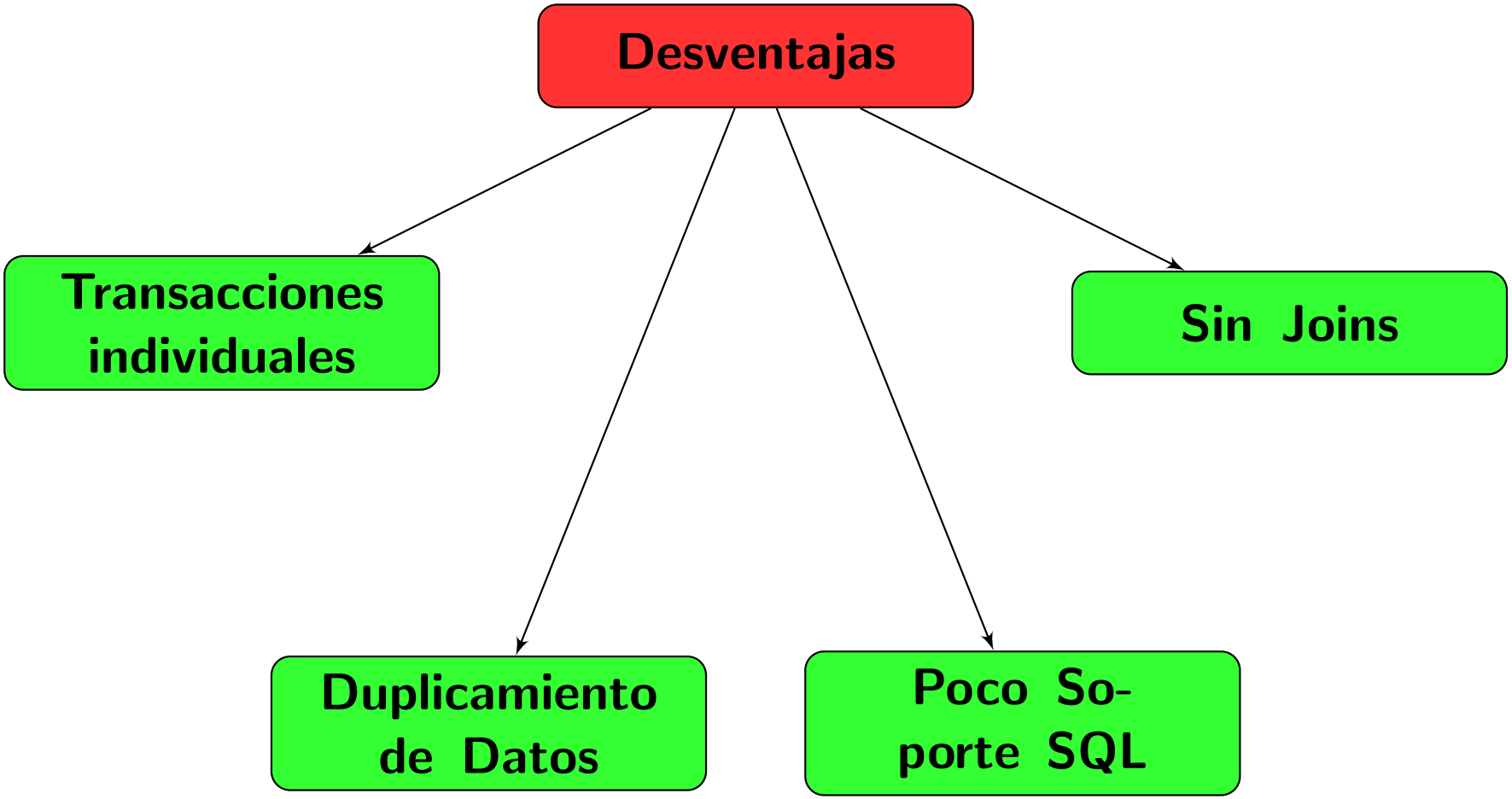
1. Soporte para computación en paralelo.
2. Actualmente: **pipelines** de MongoDB



Ventajas de MongoDB

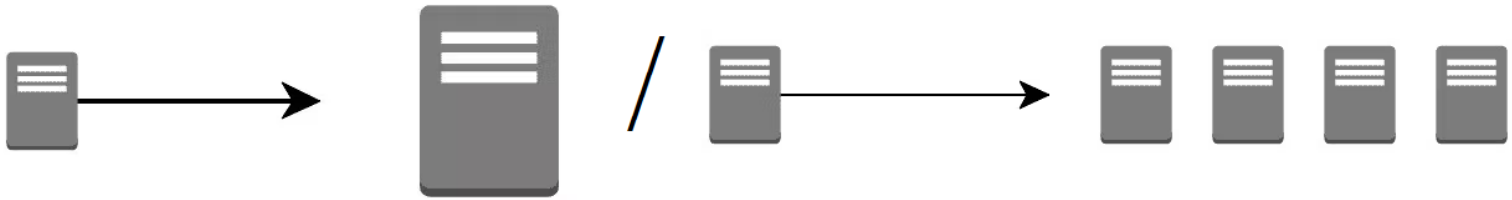


Desventajas de MongoDB



Escalamiento, Sharding y Réplica

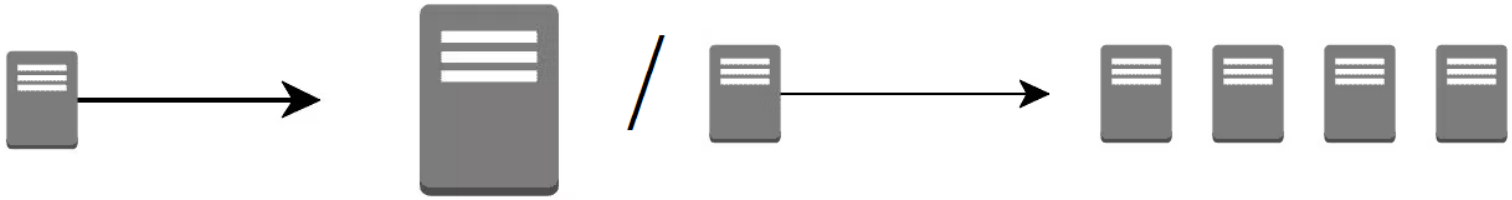
1. Escalamiento vertical/Escalamiento horizontal



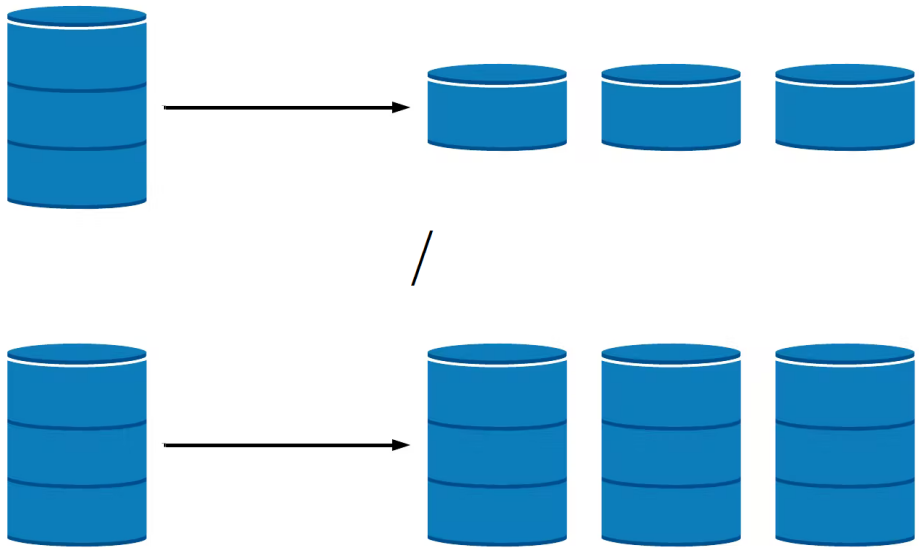
2. Sharding/Réplica

Escalamiento, Sharding y Réplica

- Escalamiento vertical/Escalamiento horizontal



- Sharding/Réplica



Ejemplo Base de Datos

```
{name: "John Smith", age: 35, city: "New York"},
{name: "Jane Doe", age: 28, city: "Los Angeles"},
{name: "Bob Johnson", age: 42, city: \Chicago"}
```

1. *Claves/Keys:* name, age, city.
2. Documento generado en MongoDB:

Ejemplo Base de Datos

```
{name: "John Smith", age: 35, city: "New York"},
{name: "Jane Doe", age: 28, city: "Los Angeles"},
{name: "Bob Johnson", age: 42, city: \Chicago"}
```

- *Claves/Keys:* name, age, city.
- Documento generado en MongoDB:

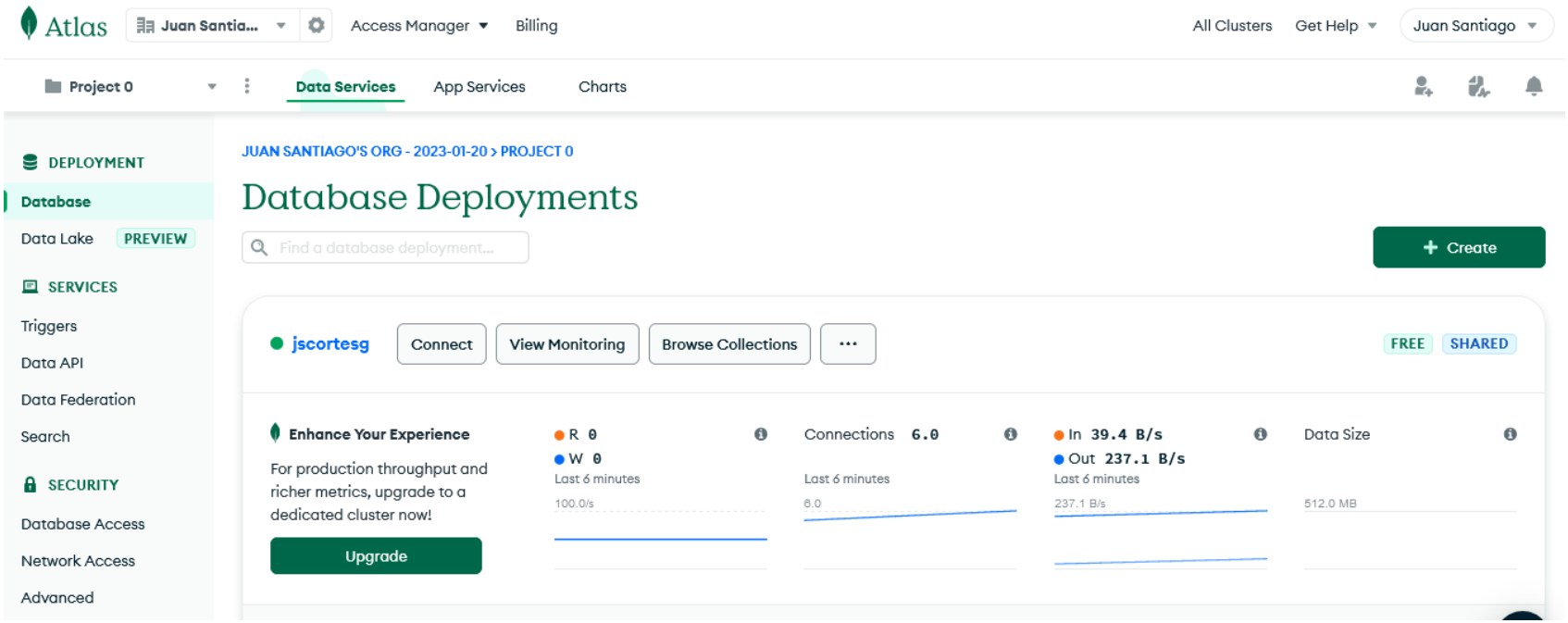
Ejemplo Base de Datos

{name: "John Smith", age: 35, city: "New York"},
{name: "Jane Doe", age: 28, city: "Los Angeles"},
{name: "Bob Johnson", age: 42, city: \Chicago"}

1. *Claves/Keys:* name, age, city.
2. Documento generado en MongoDB:

```
{
  "_id": ObjectId("6141c7315e8d5020082f97b2"),
  "name": "John Smith",
  "age": 35,
  "city": "New York"
}
```

MongoDB Atlas

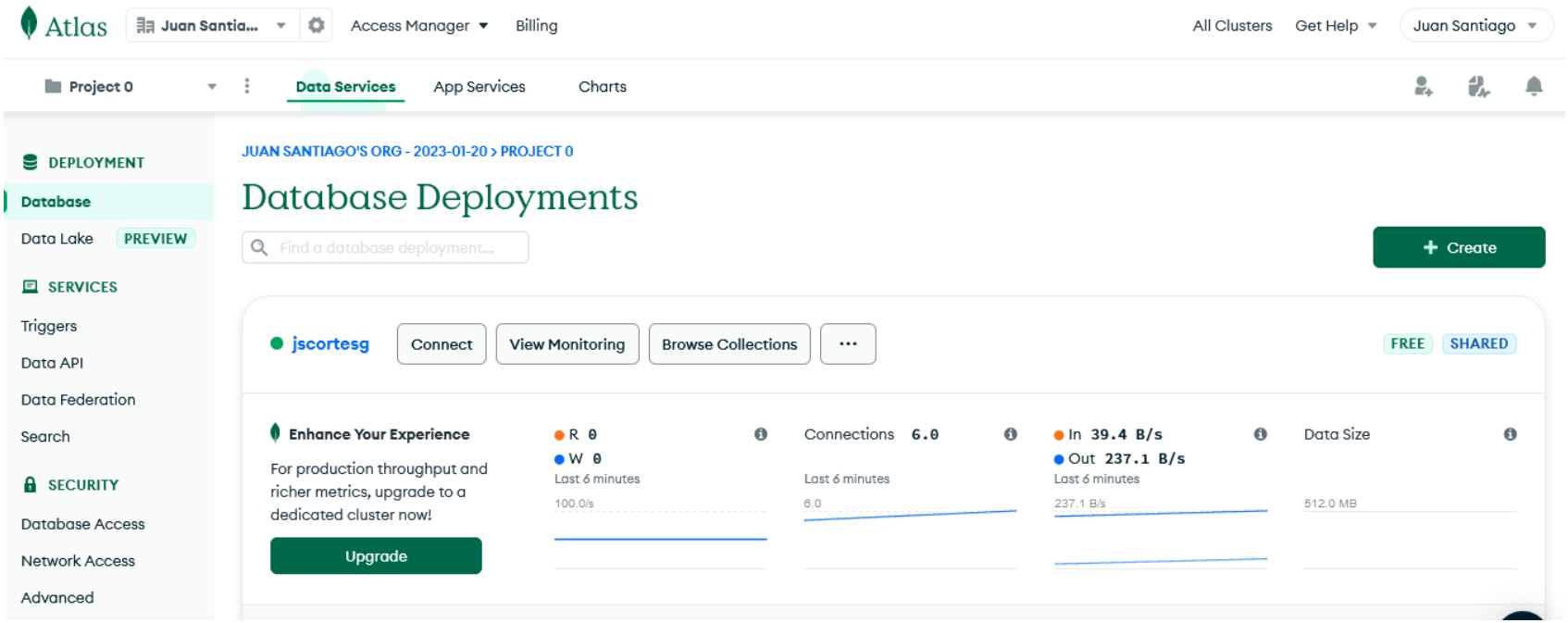


I. Ambiente para almacenar y administrar bases de datos.

II. Creación de clusters de DBs.

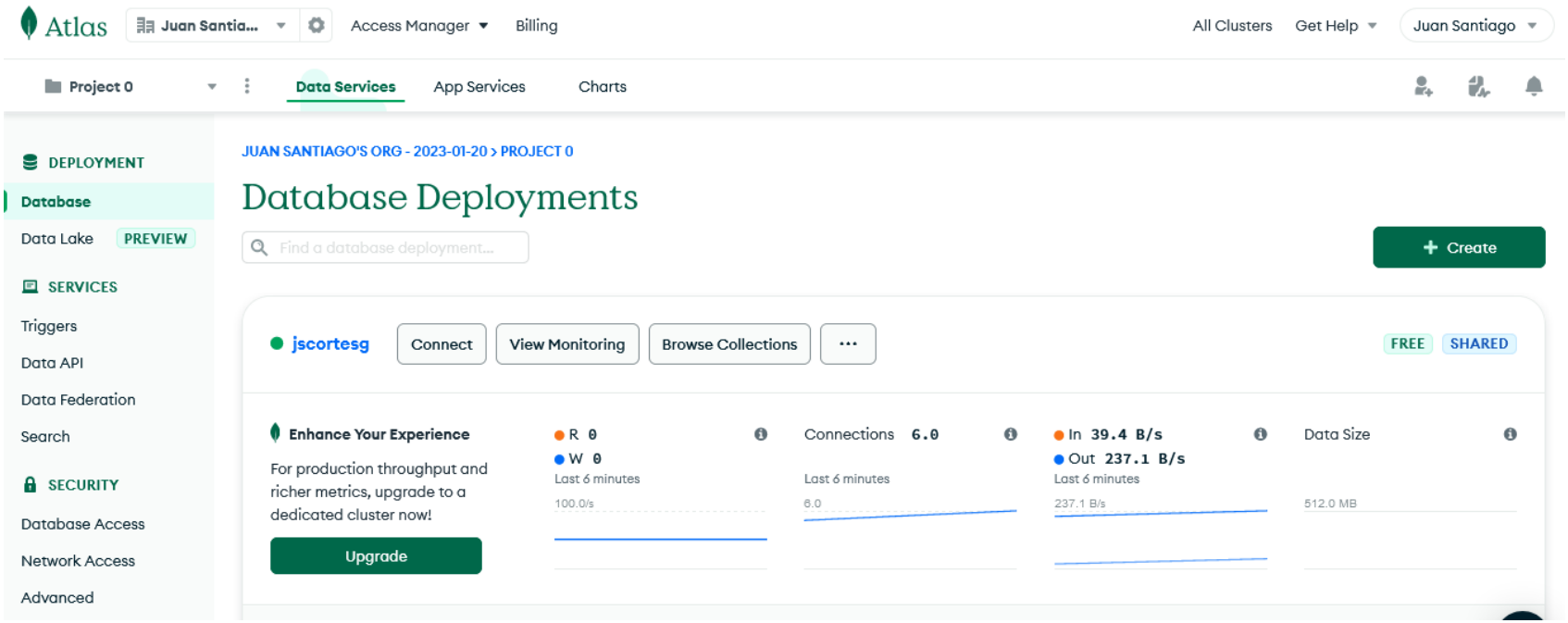
III. Conexión a MongoDB Shell.

MongoDB Atlas



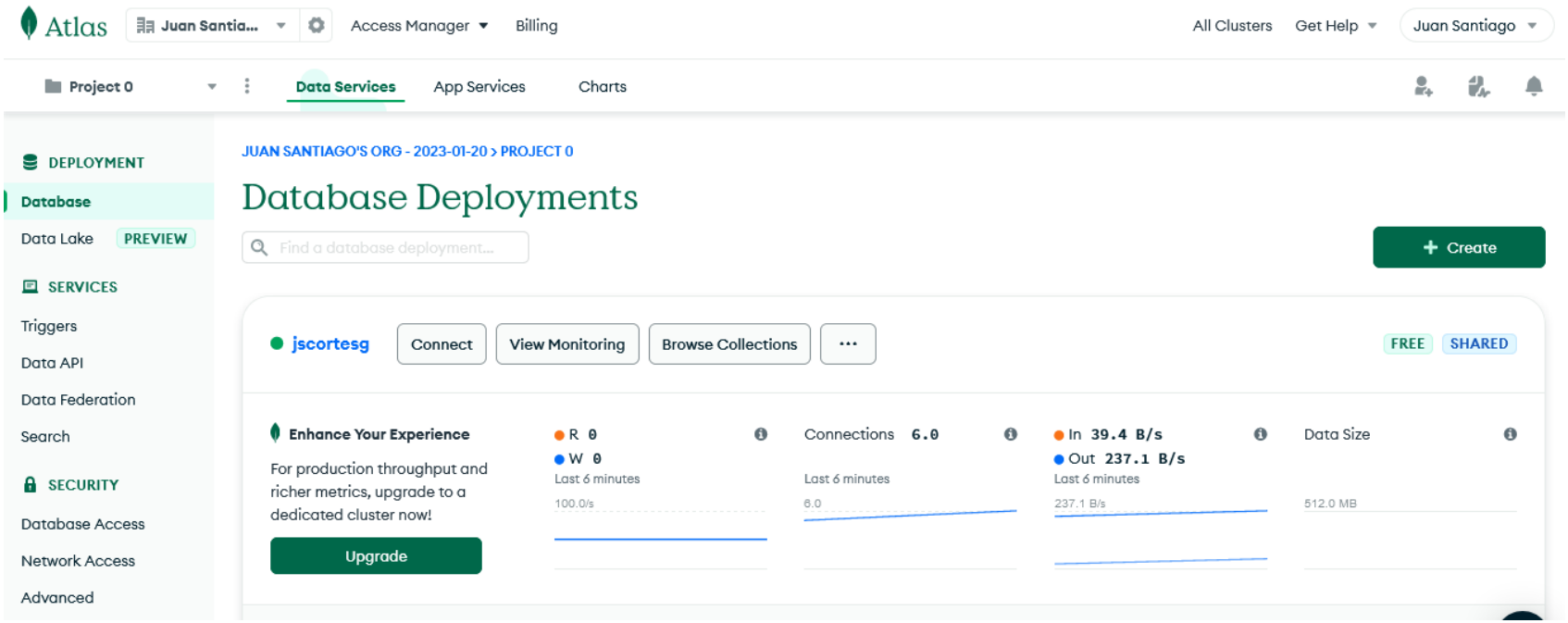
1. Ambiente para almacenar y administrar bases de datos.
2. Creación de clusters de DBs.
3. Conexión a MongoDB Shell.

MongoDB Atlas



1. Ambiente para almacenar y administrar bases de datos.
2. Creación de clusters de DBs.
3. Conexión a MongoDB Shell.

MongoDB Atlas

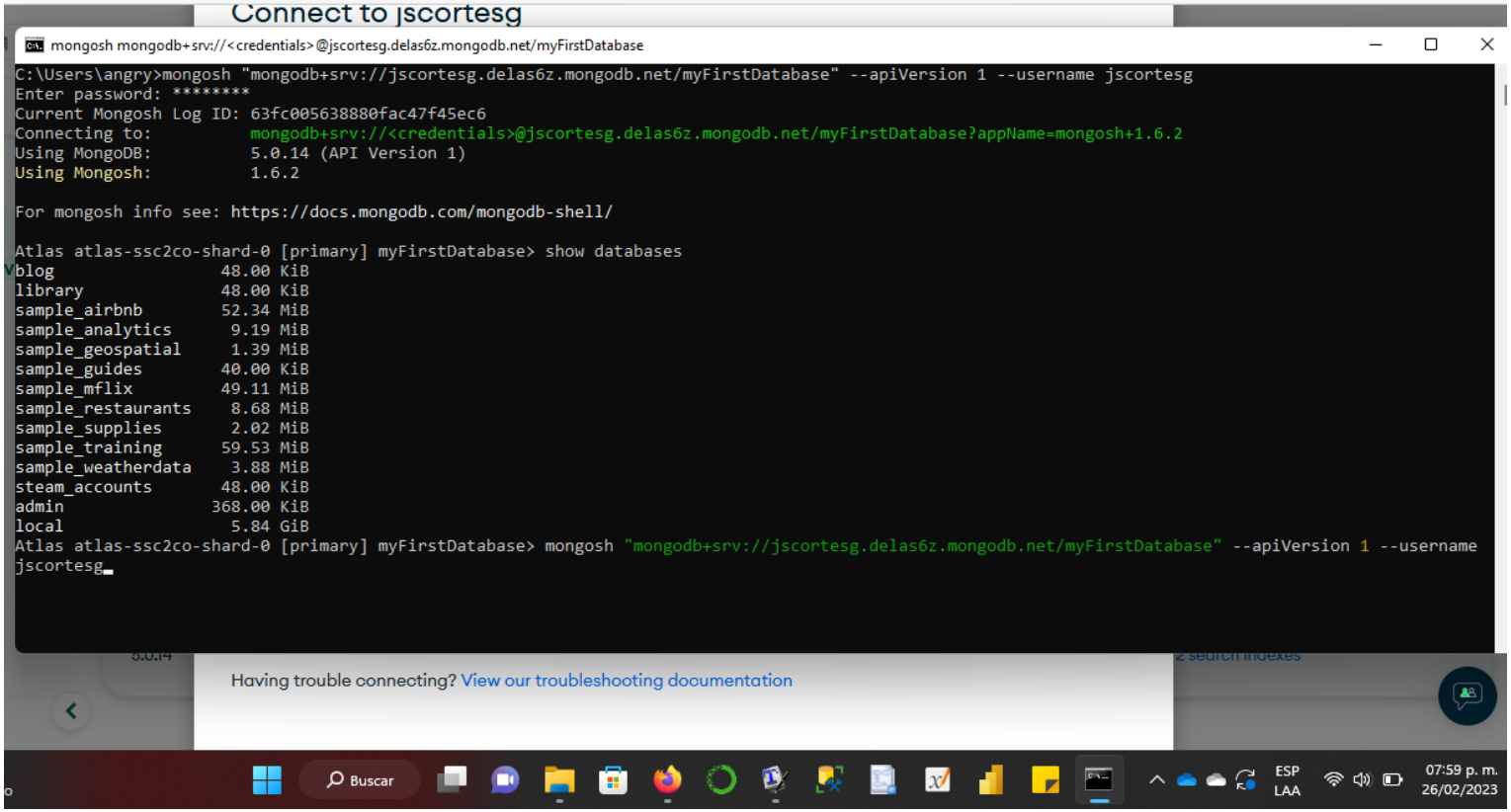


I. Ambiente para almacenar y administrar bases de datos.
II. Creación de clusters de DBs.
III. Conexión a MongoDB Shell.

Características de MongoDB Atlas

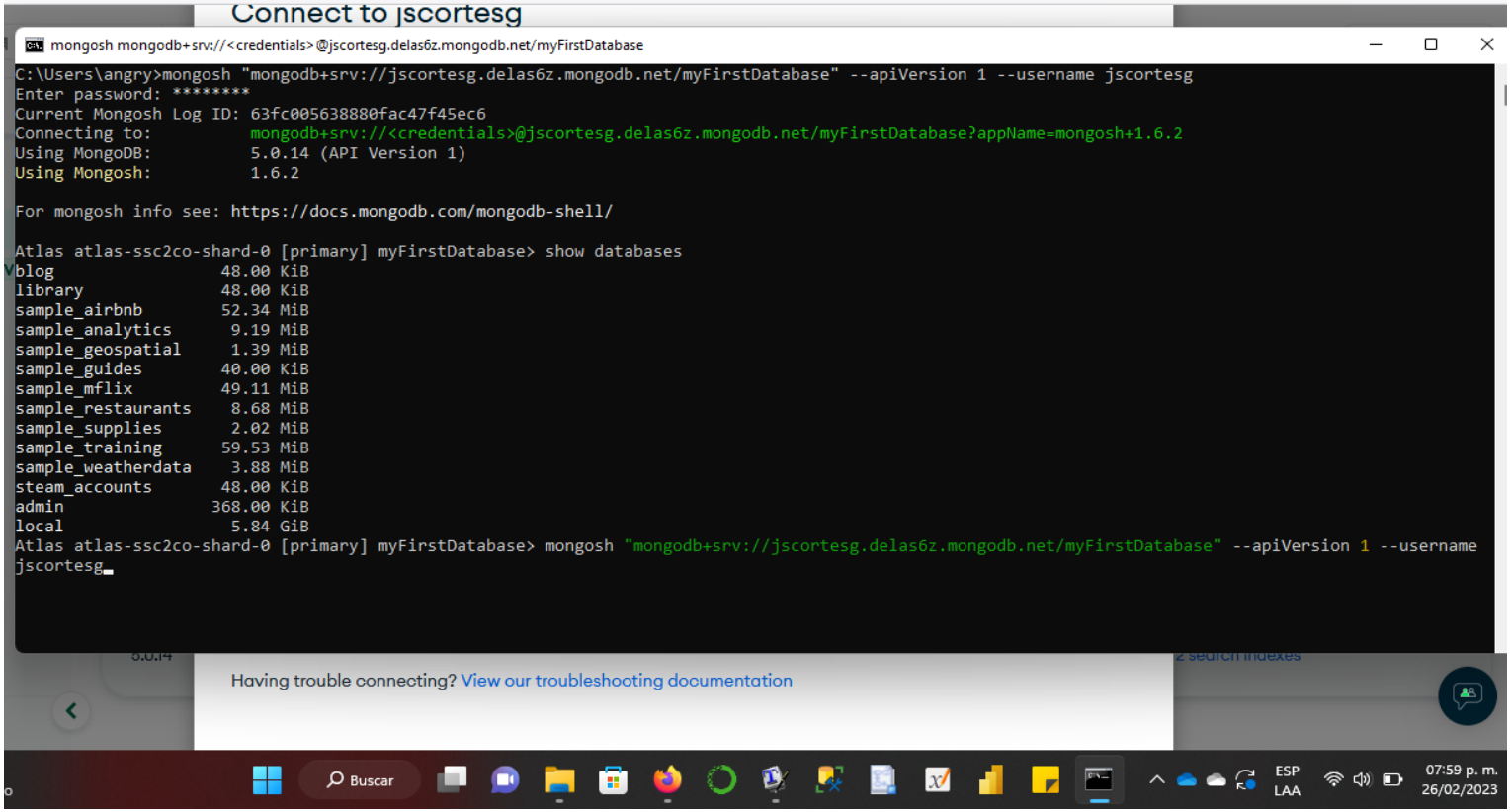
Característica	**Descripción**
Escalabilidad horizontal	Adición o eliminación de nodos.
Respaldo y recuperación	Backup y recuperación de datos automáticas.
Seguridad y cumplimiento normativo	Cifrado de datos, autenticación de usuario, etc.
Monitoreo y alertas	Detección de problemas de rendimiento . y capacidad de los clusters

MongoDB Atlas



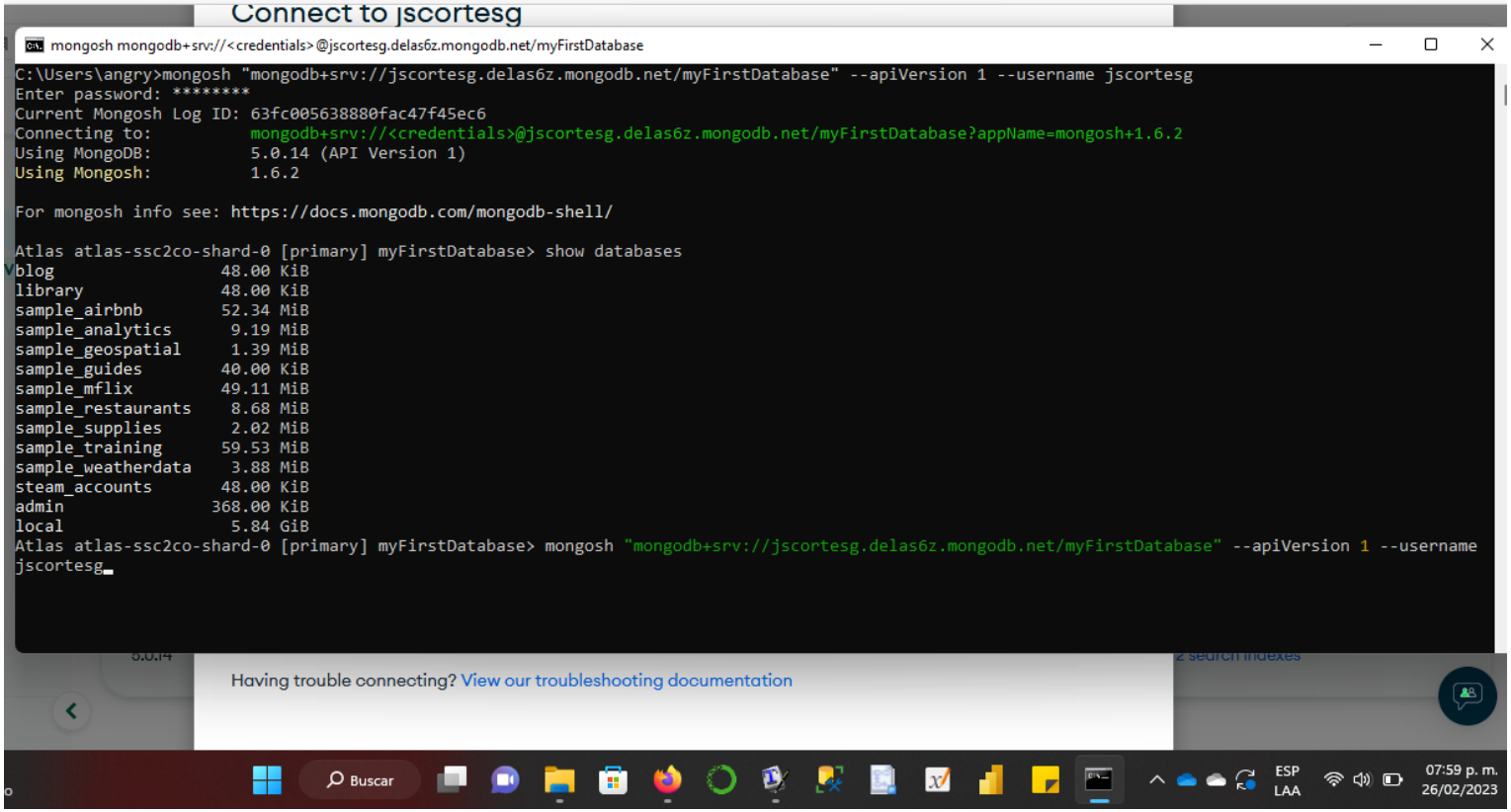
1. Interfaz de línea de comandos.
2. Ejecución de comandos en tiempo real para crear y administrar DBs.

MongoDB Atlas



1. Interfaz de línea de comandos.
2. Ejecución de comandos en tiempo real para crear y administrar DBs.

MongoDB Atlas

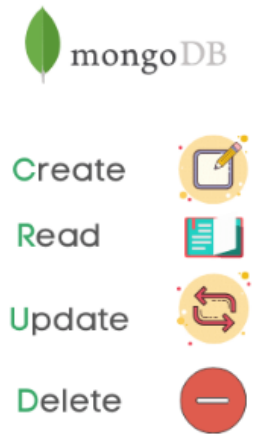


- Interfaz de línea de comandos.
- Ejecución de comandos en tiempo real para crear y administrar DBs.

Características de MongoDB Atlas

Característica	**Descripción**
Interfaz de línea de comandos	Proporciona una manera interactiva de ejecutar comandos de MongoDB y consultar datos.
JavaScript integrado	Ejecución de comandos y consulta de datos en tiempo real.
Funcionalidad de autocompletado	Facilita la escritura de comandos y consultas.
Acceso remoto a bases de datos	Conexión con una instancia de MongoDB.

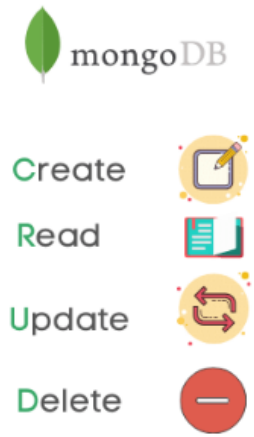
CRUD



1. *Create:* insert(), insertOne()
2. *Read:* find(), findOne()
3. *Update:* update(), updateOne()
4. *Delete:* remove(), deleteOne()
5. **Sintaxis general:**

$$\text{db.nombrebase.función()} \tag{1}$$

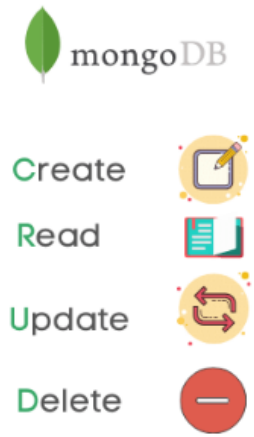
CRUD



1. *Create:* insert(), insertOne()
2. *Read:* find(), findOne()
3. *Update:* update(), updateOne()
4. *Delete:* remove(), deleteOne()
5. **Sintaxis general:**

$$\text{db.nombrebase.función()} \tag{1}$$

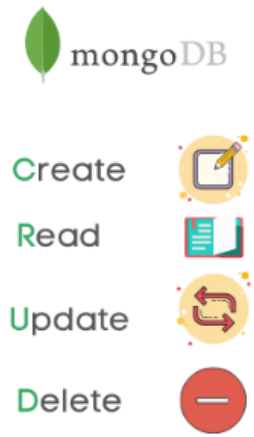
CRUD



1. *Create:* insert(), insertOne()
2. *Read:* find(), findOne()
3. *Update:* update(), updateOne()
4. *Delete:* remove(), deleteOne()
5. **Sintaxis general:**

$$\text{db.nombrebase.función()} \quad (1)$$

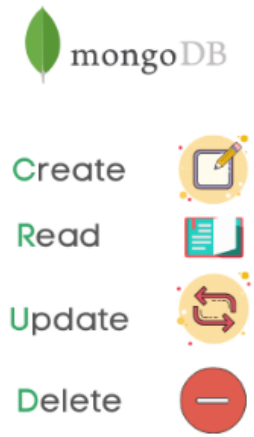
CRUD



I. *Create:* insert(), insertOne()
II. *Read:* find(), findOne()
III. *Update:* update(), updateOne()
IV. *Delete:* remove(), deleteOne()
V. **Sintaxis general:**

db.nombrebase.función() (1)

CRUD



I. *Create:* insert(), insertOne()
II. *Read:* find(), findOne()
III. *Update:* update(), updateOne()
IV. *Delete:* remove(), deleteOne()
V. **Sintaxis general:**

db.nombrebase.función() (1)

Ejemplos de Queries

I. Filtrar:

```
db.users.find({ age: { $gte: 25 } })
```

II. Ordenar:

```
db.users.find().sort({ age: -1 })
```

−1: orden **descendente**.

III. Paginación (10 documentos por página):

```
db.users.find().skip(10).limit(10)
```

IV. Métodos combinados:

```
db.users.find({ age: { $gte: 25 } })
        .sort({ last_name: -1 }).skip(10).limit(10)
```

Ejemplos de Queries

I. Filtrar:

```
db.users.find({ age: { $gte: 25 } })
```

II. Ordenar:

```
db.users.find().sort({ age: -1 })
```

-1: orden **descendente**.

III. Paginación (10 documentos por página):

```
db.users.find().skip(10).limit(10)
```

IV. Métodos combinados:

```
db.users.find({ age: { $gte: 25 } })
        .sort({ last_name: -1 }).skip(10).limit(10)
```

Ejemplos de Queries

I. Filtrar:

```
db.users.find({ age: { $gte: 25 } })
```

II. Ordenar:

```
db.users.find().sort({ age: -1 })
```

-1: orden **descendente**.

III. Paginación (10 documentos por página):

```
db.users.find().skip(10).limit(10)
```

IV. Métodos combinados:

```
db.users.find({ age: { $gte: 25 } })
        .sort({ last_name: -1 }).skip(10).limit(10)
```

Ejemplos de Queries

I. Filtrar:

```
db.users.find({ age: { $gte: 25 } })
```

II. Ordenar:

```
db.users.find().sort({ age: -1 })
```

-1: orden **descendente**.

III. Paginación (10 documentos por página):

```
db.users.find().skip(10).limit(10)
```

IV. Métodos combinados:

```
db.users.find({ age: { $gte: 25 } })
        .sort({ last_name: -1 }).skip(10).limit(10)
```

Ejemplos de Agregaciones

I. Agrupar:

```
db.products.aggregate([
   { $group: { _id: "$category", count: { $sum: 1 } } }
])
```

II. Filtrar:

III. Ordenar:

IV. Proyectar:

Ejemplos de Agregaciones

I. Agrupar:

```
db.products.aggregate([
   { $group: { _id: "$category", count: { $sum: 1 } } }
])
```

II. Filtrar:

```
db.products.aggregate([
   { $match: { category: "Electronics" } },
   { $group: { _id: null, avg_price: { $avg: "$price" } } }
])
```

III. Ordenar:

IV. Proyectar:

Ejemplos de Agregaciones

I. Agrupar:

```
db.products.aggregate([
   { $group: { _id: "$category", count: { $sum: 1 } } }
])
```

II. Filtrar:

```
db.products.aggregate([
   { $match: { category: "Electronics" } },
   { $group: { _id: null, avg_price: { $avg: "$price" } } }
])
```

III. Ordenar:

```
db.products.aggregate([
   { $match: { category: "Electronics" } },
   { $group: { _id: null, avg_price: { $avg: "$price" } } },
   { $sort: { avg_price: -1 } }
])
```

IV. Proyectar:

Ejemplos de Agregaciones

I. Agrupar:

```
db.products.aggregate([
   { $group: { _id: "$category", count: { $sum: 1 } } }
])
```

II. Filtrar:

```
db.products.aggregate([
   { $match: { category: "Electronics" } },
   { $group: { _id: null, avg_price: { $avg: "$price" } } }
])
```

III. Ordenar:

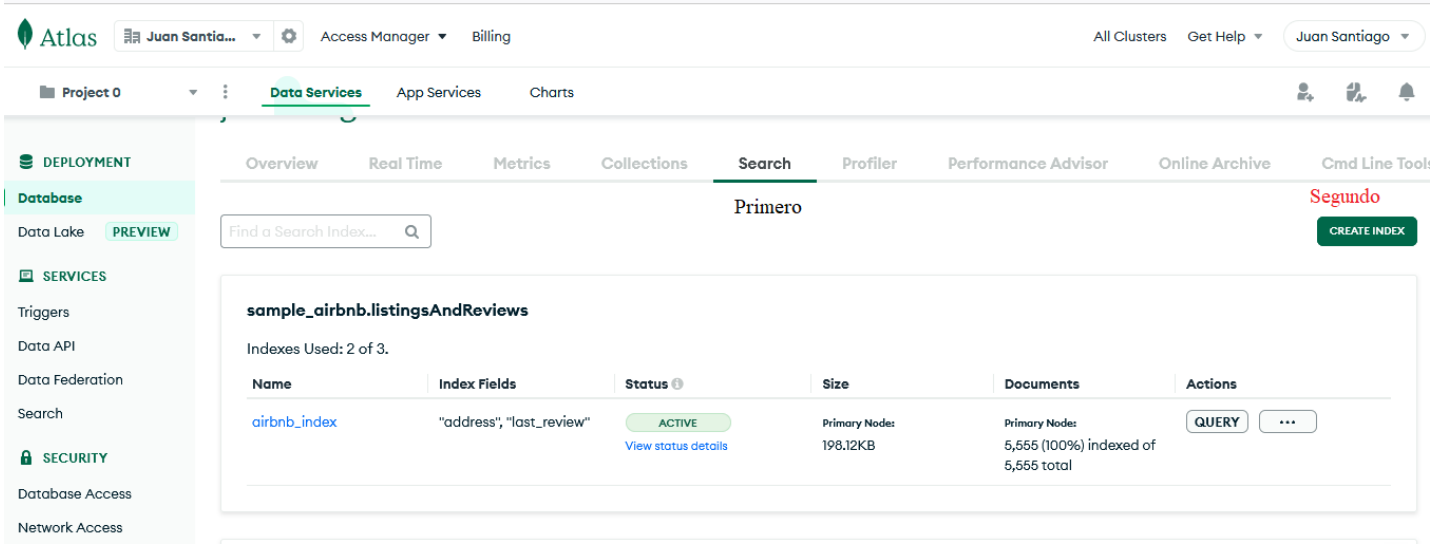
```
db.products.aggregate([
   { $match: { category: "Electronics" } },
   { $group: { _id: null, avg_price: { $avg: "$price" } } },
   { $sort: { avg_price: -1 } }
])
```

IV. Proyectar:

```
db.sales.aggregate([
   { $group: { _id: "$product", revenue: { $sum: "$price" } } },
   { $project: { name: "$_id", revenue: 1, _id: 0 } }
])
```

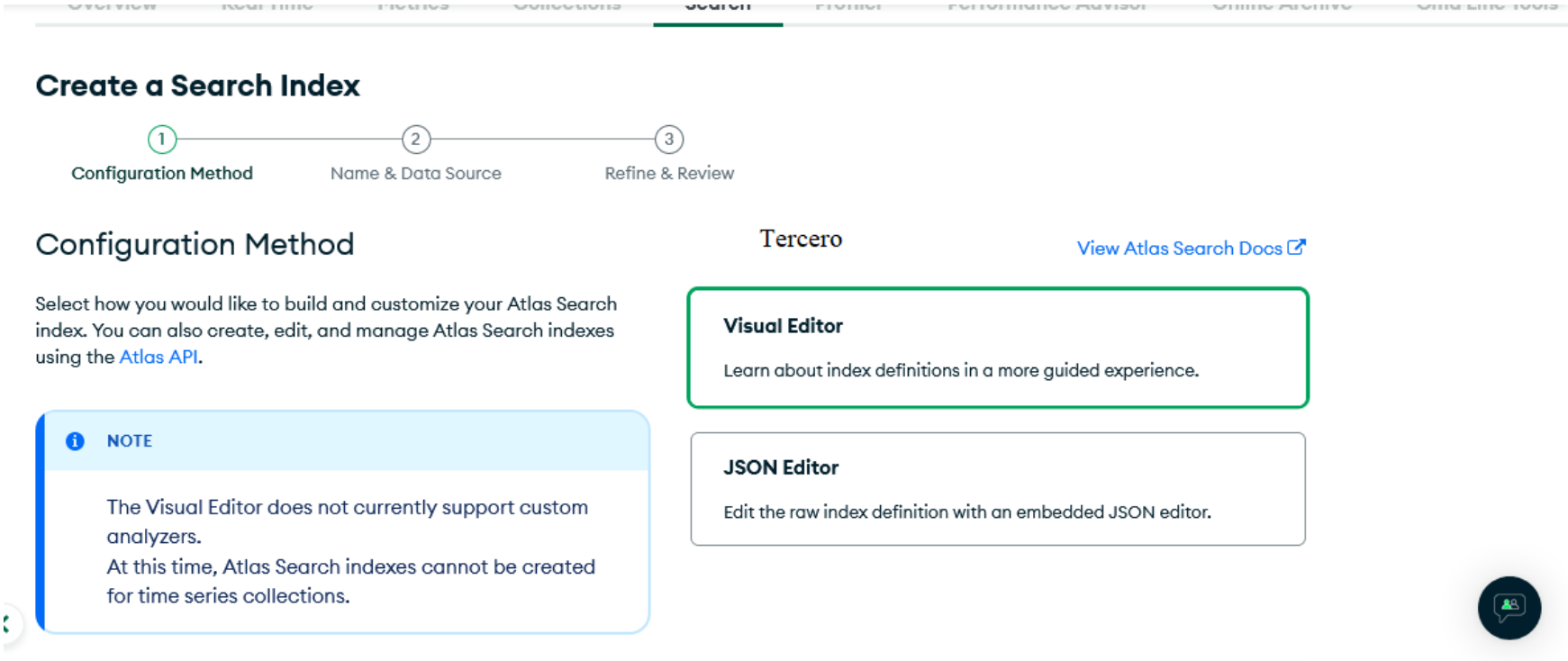
Atlas y Búsqueda

- Buscar en la DB:



Atlas y Búsqueda

1. Crear índice:



Atlas y Búsqueda

- Refinar índice: